1. Specificarea minilimbajului de programare

Tipuri de date:

* Integer: tip de date ce reprezinta numerele intregi
* Float: tip de date ce reprezinta numerele reale
* Vectori

Instructiuni:

* Instructiune de atribuire: variabila=expresie
* Instructiune de intrare/iesire: cin>>variabila;/cout<<expresie;
* Instructiune de selectie(conditionala): if(conditie){instructiuni} else{instructiuni}
* Instructiune de ciclare: while(conditie){instructiuni}

Restrictii:

* Identificatorii pot continue doar litere, cifre si caracterul “\_”
* Identificatorii trebuie sa inceapa cu o litera
* Identificatorii nu trebuie sa depaseasca 8 caractere
* Constantele de tip float trebuie sa contina caracterul “.”

Clasificarea atomilor

1. Identificatori

ID ::= litera (litera|cifra|”\_”) {0,7}

litera ::= “A”|”B”|…|”Z”|”a”|”b”|…|”z”

cifra ::= ”0”|“1”|”2”|“3”|”4”|”5”|”6”|”7”|”8”|”9”

1. Constante

CONST ::= “0” | ([“+”|”-”] cifra\_nenula (cifra\_nenula|”0”)\* [ “.” (cifra\_nenula|”0”)\* cifra\_nenula ])

CONST\_POZ ::= cifra\_nenula (cifra\_nenula|”0”)\* [ “.” (cifra\_nenula|”0”)\* cifra\_nenula ]

CONST\_STRING ::= “”” {cifra | litera | caracter\_special} “””

cifra ::= ”0”|“1”|”2”|“3”|”4”|”5”|”6”|”7”|”8”|”9”

litera ::= “a”|”b”|”c”| … | “z” |”A”|”B”|…|”Z”

caracter\_special ::= ”!”|”%”|”^”|”&”|”\*“|”(“|”)”|”\_”|”-”|”+”|”=”|

”[”|”{“|”]”|”}”|”;”|”:”||”,”|”<”|”.”|”>”|”/”

cifra\_nenula ::= “1”|”2”|“3”|”4”|”5”|”6”|”7”|”8”|”9”

3. Cuvinte rezervate

“int”, “float”, “void”, “main”, “cout”, “cin”, “while”, “if”, “else”, “endl”

4. Operatori

“+”, “-”, “\*”, “<<”, “>>”, “=”, “!=”, “> ”, ”<”, “<=”, “>=”, “==”, “[”, “]”

5. Separatori

“(”, “)”, “{”, “}”, “,”, “;”, “ ”, “ “ ”

Specificarea minilimbajului de programare folosind EBNF

program ::= antet\_biblioteca”\n using namespace std;”{ lista\_functii}

antet\_biblioteca::= “#” ”include” ”<” ”iostream” ”>”

lista\_functii ::= “int” “main” “(” “)” “{” [lista\_decl] [lista\_instr] “}”

lista\_decl ::= declarare “;” {declarare “;”}

declarare ::= tip lista\_def

tip ::= (“int” | “float”) [ “[” CONST\_POZ|ID “]” ]

lista\_def ::= definire {“,” definire}

definire ::= ID [ “=” CONST ]

lista\_instr ::=instr {“;” instr}

instr ::= atribuire | instr\_if | instr\_loop | intrare | iesire

atribuire ::= ID “=” variabila { operator\_ aritm variabila}

variabila ::= ID | CONST | variabila\_compusa

variabila\_compusa ::= ID “[” (CONST\_POZITIV | ID) “]”

operator\_aritm ::= “\*”|”-”|”+”

instr\_if ::= “if” ”(” conditie “)” “{” lista\_instr “}” [“else” “{” lista\_instr “}” ]

conditie ::= variabila operator\_rel variabila

operator\_rel ::= “!=”|“==”|“> ”|”<”|“<=”|“>=”

instr\_loop ::= “while” “(” conditie “)” “{” lista\_instr ”}”

intrare ::= “cout” scriere {scriere}

scriere ::= “<<” ( variabila | “endl” | CONST\_STRING )

iesire ::= “cin” “>>” ID

1. Se cer textele sursa a 3 mini-programe

#include <iostream>

using namespace std;

int main(){

int R = 3;

float arie, perimetru;

arie = 3.141 \* R \* R;

perimetru = 2 \* 3.141 \* R;

cout << "Aria: " << arie << endl;

cout << "Perimetrul: " << perimetru << endl;

}

#include <iostream>

using namespace std;

int main()

{

int a, b;

cin>>a;

cin>>b;

while(a != b)

{

if(a > b)

a = a - b;

else

b = b - a;

}

cout << "CMMDC:" << a << endl;

}

#include <iostream>

using namespace std;

int main()

{

float nums[100], sum = 0;

int n, i = 0;

cout << "n="; cin>>n;

while (i < n)

{ cin >> nums[i];

sum = sum + nums[i];

i = i + 1;

}

cout << "sum: " << sum << endl;

}

3.Se cer textele sursa a 2 programe care contin erori

A. doua erori in limbajul original (inclusive MLP)

Int main()

{

float nums[100], sum = 0;

int n, i = 0 //->eroare: fara ; dupa decl

cin>>n;

while (i < ) // ->eroare : fara a doua variabila din conditie

{ suma = suma + i;

i = i + 1;

}

cout << suma << endl;

}

A. doua erori in MPL (fara limbajul original)

Int main()

{

float arrayOfNums[100], sum = 0; //err -> ID cu lungime //mai mare de 8 caractere

int n, i = 0;

cout << "n="; cin>>n;

for( i = 0; i < n ; i++) // ->err : nu exista for in MLP

{ cin >> nums[i];

sum = sum + nums[i];

i = i + 1;}

cout << "sum: " << sum << endl;}